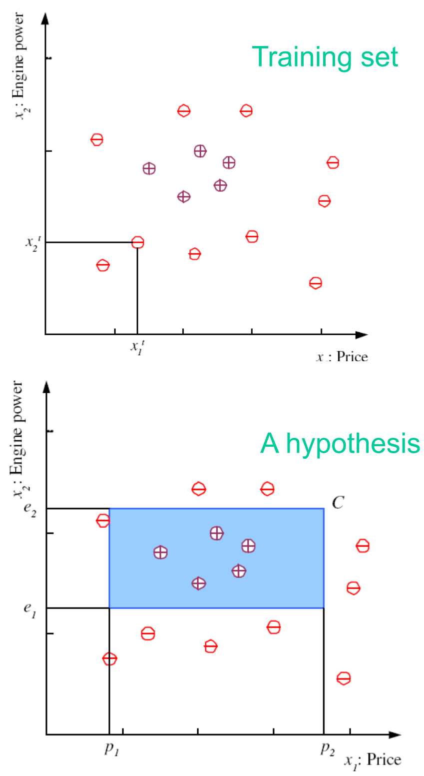
Supervised Learning

- Learning a Class from Examples
- Vapnik-Chervonkis Dimension
- Probably Approximately Correct Learning
- Noise and Model Complexity
- Multiple Classes
- Regression
- Model Selection and Generalization

Learning a Class From Examples (1)

- Problem
  - To determine whether a car $\mathbf{x}$ is a family car
- Training set $D = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)\}$
  - Input $\mathbf{x} = [x_1, x_2]$
    - x_1: price
    - x_2: engine power
  - Output y
    - $y = \begin{cases} 1, \mathbf{x} \text{ is a positive example (family cars)} \\ 0, \mathbf{x} \text{ is a negative example (the other cars)} \end{cases}$
- Unknown function $C: X \to Y$
  - Ideal class function for family cars
    - $y_i = C(\mathbf{x}_i)$



Figs. 2.1 – 2.2 [Alpaydin 20]

Learning a Class From Examples (2)

- The hypothesis set H (set of candidate functions)
  - Assume C to be a rectangle in the price-engine power space
    - $(p_1 \leq price \leq p_2)\ AND\ (e_1 \leq engine\ power \leq e_2)$
  - H: the set of all possible rectangles
  - A particular hypothesis $h \in H$
    - Specified by a particular quadruple $(p_1^h, p_2^h, e_1^h, e_2^h)$
    - $h(\mathbf{x}) = \begin{cases} 1, \text{if } h \text{ classifies } \mathbf{x} \text{ as a positive example} \\ 0, \text{if } h \text{ classifies } \mathbf{x} \text{ as a negative example} \end{cases}$
- The learning algorithm
  - To find a $h \in H$ that is as similar as possible to C
    - The learning problem reduces to finding the 4 parameters
    - But we do not know C, how can we evaluate how well $h(\mathbf{x})$ matches $C(\mathbf{x})$?

Learning a Class From Examples (3)

- Empirical error
  - The error of hypothesis h given the training set D
  - $E(h|D) = \sum_{i=1}^{N} 1(h(\mathbf{x}_i) \neq y_i)$
    - $1(a \neq b) = \begin{cases} 1, \text{if } a \neq b \\ 0, \text{if } a = b \end{cases}$

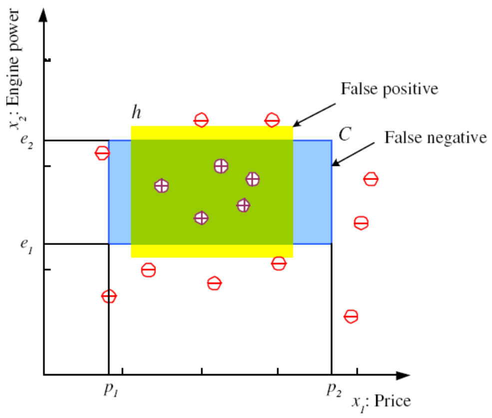


Fig. 2.3 [Alpaydin 20]

		Actual Label	
		P	N
Test Result	P	True Positive (Hit)	False Positive (False Alarm)
	N	False Negative (Miss)	True Negative (Correct Rejection)

Learning a Class From Examples (4)

- Generalization
  - How well a hypothesis will correctly classify future examples
- Version space
  - The subset of H that is consistent with the training set D
- Examples
  - $h = S$
    - The tightest rectangle that includes all the positive examples and none of the negative examples
  - $h = G$
    - The largest rectangle with no error
  - Any $h \in H$ between S and G
    - Make up the version space

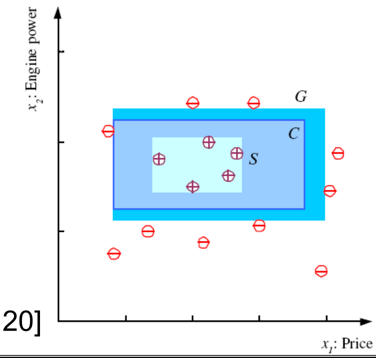


Fig. 2.4 [Alpaydin 20]

Learning a Class From Examples (5)

- Consistent hypothesis $h \in H$
  - Not unique
  - Additional constraint?
    - e.g., margin
      - The distance between the boundary and the instances closest to it
- Maximal margin classifier
  - To seek h with the maximum margin
    - h is determined by a subset of instances
      - Support vectors
    - Other instances can be removed without affecting h

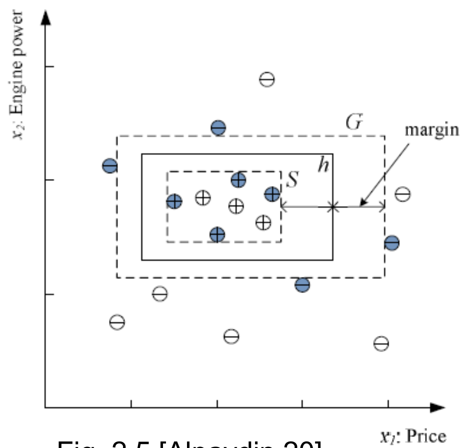


Fig. 2.5 [Alpaydin 20]

VC Dimension (1)

- Goal
  - To relate the size N of the training data set with the generalization performance of the classifier
- In a binary classification problem
  - Given a dataset D containing N points $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$
    - These points can be labeled in 2^N ways as positive and negative
      - 2^N different learning problems can be defined
      - $H(D) = \{+, -\}^N$
    - The max number of dichotomies (partitions) on a set of N points is 2^N
      - Two different $h's$ may generate the same dichotomies
- Shattering
  - If for any of these 2^N problems
    - We can find a $h \in H$ that separates $+$ from $-$
  - Then we say H shatters the N points $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$

VC Dimension (2)

- Example
  - In 2-D space $\mathbf{x} \in \{0,1\}^2$
    - If we include all possible combinations in D
    - $N = 4$
      - These 4 points can be labeled in $2^4 = 16$ ways as positive and negative
      - $$\left\{\begin{matrix} \begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}, \begin{bmatrix}0 & 0\\0 & 1\end{bmatrix}, \begin{bmatrix}0 & 0\\1 & 0\end{bmatrix}, \begin{bmatrix}0 & 0\\1 & 1\end{bmatrix}, \begin{bmatrix}0 & 1\\0 & 0\end{bmatrix}, \begin{bmatrix}0 & 1\\0 & 1\end{bmatrix}, \begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}, \begin{bmatrix}0 & 1\\1 & 1\end{bmatrix} \\ \begin{bmatrix}1 & 0\\0 & 0\end{bmatrix}, \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}, \begin{bmatrix}1 & 0\\1 & 0\end{bmatrix}, \begin{bmatrix}1 & 0\\1 & 1\end{bmatrix}, \begin{bmatrix}1 & 1\\0 & 0\end{bmatrix}, \begin{bmatrix}1 & 1\\0 & 1\end{bmatrix}, \begin{bmatrix}1 & 1\\1 & 0\end{bmatrix}, \begin{bmatrix}1 & 1\\1 & 1\end{bmatrix} \end{matrix}\right\}$$
  - Let H be the set of linear function in 2-D
    - Then the set of linear function can form only 14 distinct dichotomies out of the 16 possibilities
    - However, this set of linear functions can form all possible 8 dichotomies for $N = 3$ points
      - $VC(H) = 3$

VC Dimension (3)

- Vapnik-Chervonkis dimension of a hypothesis set H
  - $VC(H)$
    - The maximum number of points that can be shattered by H
      - Measuring the capacity of H

- Example (Fig. 2.6)
  - H: the set of axis-aligned rectangles in 2-D
    - $VC(H) = 4$
    - There is a set of 4 points that can be shattered by H
    - But no 5 points can be shattered by H
  - With H, we can learn only datasets containing 4 points

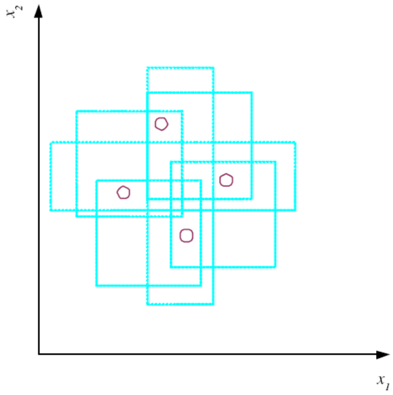


Fig. 2.6 [Alpaydin 20]

VC Dimension (4)

- The two examples shows that we can learn only datasets containing small number of points
  - Linear functions H in $d-dimensional$ space
    - $VC(H) = d + 1$
    - H can shatter at most $(d + 1)$ samples
  - Rectangular functions H in $d-dimensional$ space
    - $VC(H) = 2d$
    - H can shatter at most $(2d)$ samples
- However, in real life
  - Neighboring data points usually have the same labels
  - May not need to consider all possible labelings
  - Good generalization performance is expected if the number of training samples is a few times the VC dimension

PAC Learning (1)

- Probably approximately correct learning
  - Given
    - A class C, and
    - Examples drawn from some unknown but fixed $p(x)$
  - To find N such that
    - With probability at least $1-\delta$, where $\delta \leq 1/2$
    - The hypothesis h has error at most $\epsilon > 0$
      - $P\{C\Delta h \leq \epsilon\} \geq 1-\delta$
        - » $C\Delta h$: the region of difference between C and h
      - The minimum number of N that guarantees, with high probability, the design of a classifier with good error performance
  - Probably
    - The probability of the bound to fail is small ($< \delta$)
  - Approximately correct
    - When the bound holds, the error is small ($< \epsilon$)

To understand how large a dataset needs to be in order to give good generalization

PAC Learning (2)

- Example
  - The hypothesis S: the tightest rectangle
  - The error region between C and $h = S$ is 4 rectangular strips
  - To make sure the probability of a positive $\oplus$ point falling in the region (i.e., an error) is at most ϵ
    - Each strip is at most $\epsilon/4$
    - Pr a random sample misses a strip is $(1-\frac{\epsilon}{4})$
    - Pr that N instances miss a strip $\left(1-\frac{\epsilon}{4}\right)^N$
    - Pr that N instances miss 4 strips $4\left(1-\frac{\epsilon}{4}\right)^N$
    - To have $4\left(1-\frac{\epsilon}{4}\right)^N \leq \delta$ and using $(1-x) \leq \exp(-x)$
    - We have $4\left(1-\frac{\epsilon}{4}\right)^N \leq 4\exp\left(-\frac{N\epsilon}{4}\right) \leq \delta \Rightarrow N \geq \left(\frac{4}{\epsilon}\right)\ln(\frac{4}{\delta})$

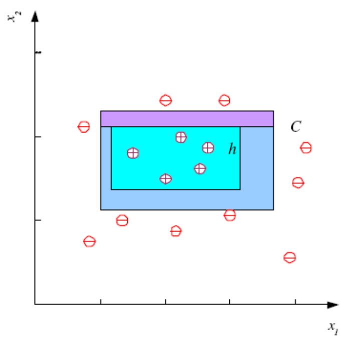


Fig. 2.7 [Alpaydin 20]

PAC Learning (3)

- The bounds derived within the PAC learning
  - Often described as worst-case bound
    - Because they apply to any choice of data distribution
- In real-world applications
  - We deal with distributions with significant regularity
  - The PAC bounds are very conservative
    - Overestimate the size of datasets required to achieve a given generalization performance

Noise and Model Complexity

- Noise (or outliers)
  - Points that are not consistent with the rest of the training data
    - Imprecision in recording
    - Errors in labeling
  - When there is noise
    - Zero error may not be possible with a simple hypothesis
- Simpler model
  - Lower computation complexity
  - Easier to train
    - Fewer parameters
  - Easier to explain
  - Generalizes better

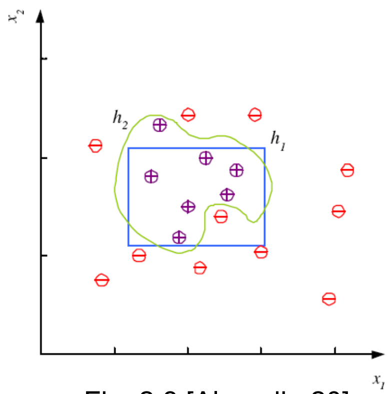


Fig. 2.8 [Alpaydin 20]

Multiple Classes (1)

- K-class problem
  - K classes: $C_1, \dots, C_K$
  - The training set $D = \{(\mathbf{x}_1, \mathbf{y}_1), \dots, (\mathbf{x}_N, \mathbf{y}_N)\}$
    - $\mathbf{y} = [y_1, \dots, y_K]$ has K dimensions
    - $y_i = \begin{cases} 1, \text{if } \mathbf{x} \in C_i \\ 0, \text{if } \mathbf{x} \in C_j, j \neq i \end{cases}$

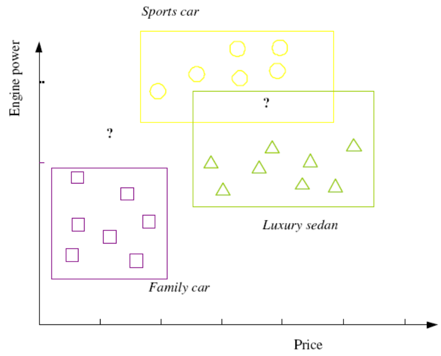


Fig. 2.9 [Alpaydin 20]: "?" are reject regions where no, or more than one, class is chosen

- One-against-the-rest
  - Use K two-class problems
    - Each h_i assigns $\mathbf{x}$ to C_i or not C_i
    - $h_i(\mathbf{x}) = \begin{cases} 1, \text{if } \mathbf{x} \in C_i \\ 0, \text{if } \mathbf{x} \in C_j, j \neq i \end{cases}$
  - Class imbalance problem
    - #negative >> #positive

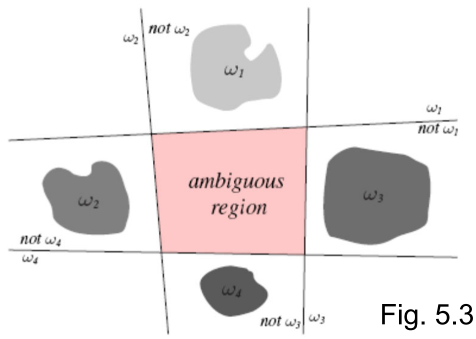


Fig. 5.3 [Duda 01]

Multiple Classes (2)

- Pairwise separation
  - Use $K(K-1)/2$ classifers
    - For every pair of classes
    - Decision can be made by majority vote

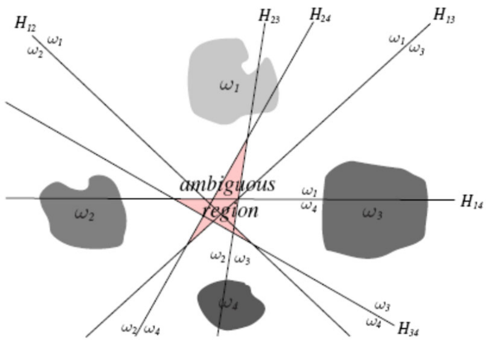


Fig. 5.3 [Duda 01]

- Use K discriminant functions (Sec. 3.4)
  - Define a set of discriminant functions $g_i(\mathbf{x}), i = 1, \dots, K$
  - $\mathbf{x} \to C_i$ if $g_i(\mathbf{x}) = \max_k g_k(\mathbf{x})$

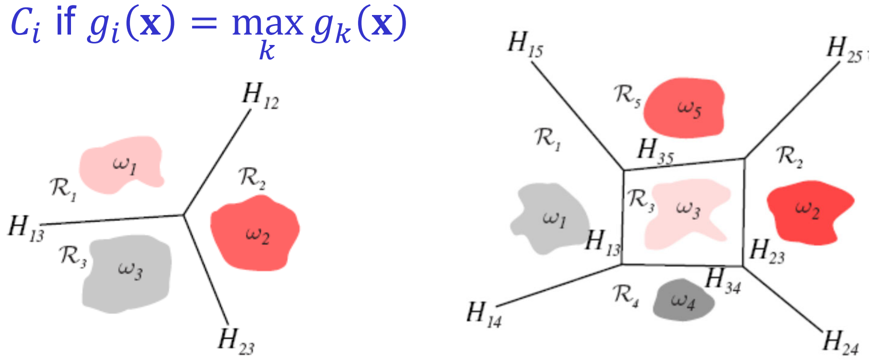
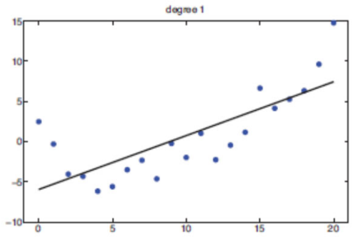


Fig. 5.4 [Duda 01]

Regression (1)

- Regression
  - $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_N, y_N)\}$
    - The output y is continuous (real-valued)
    - $y = f(\mathbf{x}) + \varepsilon,\ \varepsilon$ is random noise
      - We often model ε using $\varepsilon \sim N(0, \sigma^2)$
        - » A Gaussian with zero mean and constant variance
  - To approximate the output by the model $y = g(\mathbf{x})$
    - By minimizing the empirical error on D
      - $E(g|D) = \frac{1}{N}\sum_{i=1}^{N}[y_i - g(\mathbf{x}_i)]^2$



(a)

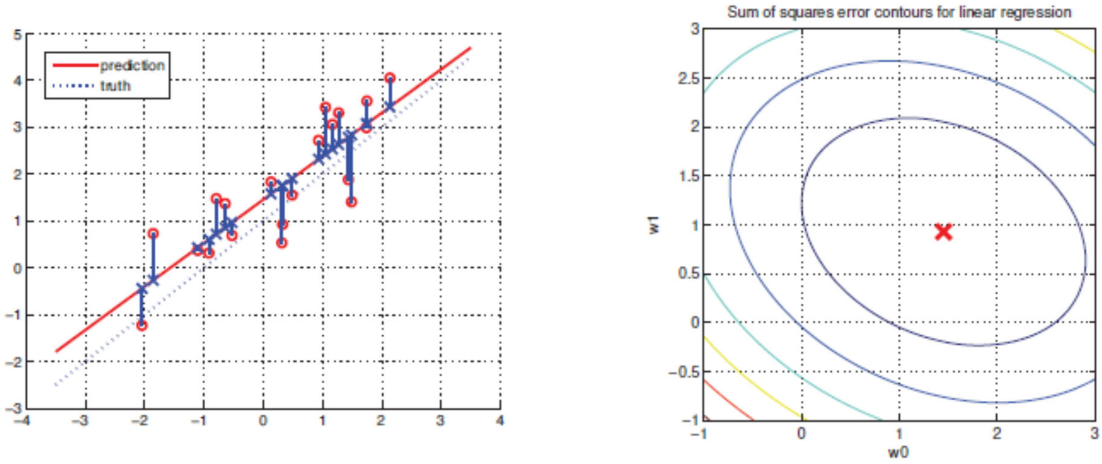
[Murphy 2012]

Figure 1.7 (a) Linear regression on some 1d data.

Regression (2)

- Linear regression
  - Assuming that $g(\mathbf{x})$ is linear
    - $g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} = w_d x_d + \cdots + w_1 x_1 + w_0$
  - To minimize
    - $E(\boldsymbol{w}|D) = \frac{1}{2}\sum_{i=1}^{N}\left(y_i - \boldsymbol{w}^T\boldsymbol{x}_i\right)^2 = \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2$
      - $\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}_1^T \\ \vdots \\ \boldsymbol{x}_N^T \end{bmatrix}$
    - Let $\frac{\partial E(\boldsymbol{w})}{\partial \boldsymbol{w}} = -\boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}) = 0$
    - We have
      - $\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w} = \boldsymbol{X}^T\boldsymbol{y}$ (normal equation)
      - $\widehat{\boldsymbol{w}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}$

Regression (3)

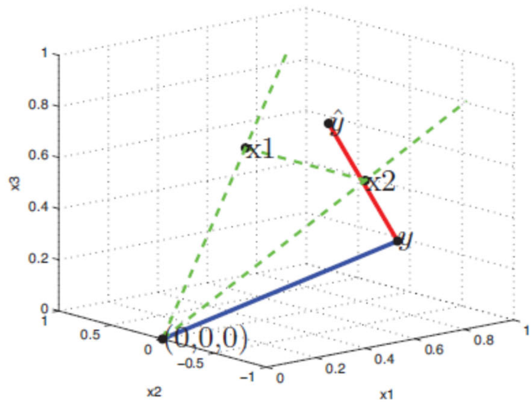


(a) (b)

Figure 7.2 (a) In linear least squares, we try to minimize the sum of squared distances from each training point (denoted by a red circle) to its approximation (denoted by a blue cross), that is, we minimize the sum of the lengths of the little vertical blue lines. The red diagonal line represents $\hat{y}(x) = w_0 + w_1 x$, which is the least squares regression line. Note that these residual lines are not perpendicular to the least squares line, in contrast to Figure 12.5. Figure generated by `residualsDemo`. (b) Contours of the RSS error surface for the same example. The red cross represents the MLE, $\mathbf{w} = (1.45, 0.93)$. Figure generated by `contoursSSEdemo`. [Murphy 2012]

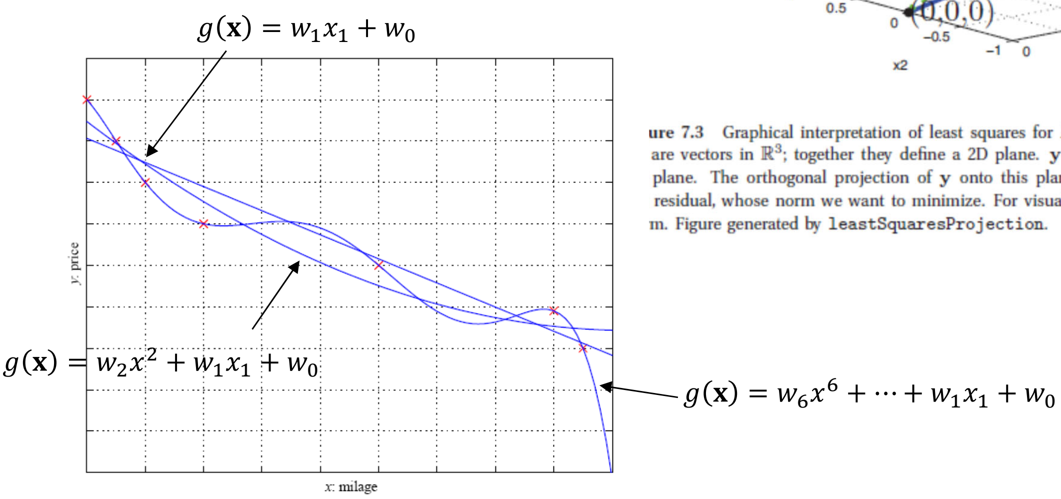
Regression (4)

- Example
  - In polynomial fitting
    - $\boldsymbol{x} = [1, x, x^2, \dots, x^d]^T$



ure 7.3 Graphical interpretation of least squares for $N = 3$ examples and $D = 2$ features. $\tilde{\mathbf{x}}_1$ and are vectors in $\mathbb{R}^3$; together they define a 2D plane. $\mathbf{y}$ is also a vector in $\mathbb{R}^3$ but does not lie on this plane. The orthogonal projection of $\mathbf{y}$ onto this plane is denoted $\hat{\mathbf{y}}$. The red line from $\mathbf{y}$ to $\hat{\mathbf{y}}$ is residual, whose norm we want to minimize. For visual clarity, all vectors have been converted to unit m. Figure generated by `leastSquaresProjection`.

[Murphy 2012]



Model Selection & Generalization (1)

- Ill-posed problem
  - The training set is only a small sample in the domain
  - No unique solution can be determined using only the available information
- Inductive bias
  - Assumptions that define the model selection criteria
- Generalization
  - How well a model trained on the training set predicts the right output for new instances
- Triple trade-off
  - Complexity of the hypothesis H
  - The amount N of training data
  - Generalization error E on new examples

Model Selection & Generalization (2)

- To estimate generalization error
  - Cross validation
    - Partitioning the training set into two (e.g., 80% training+ 20% validation)
      - Training set
        - » To fit the hypothesis
          - » e.g., the coefficients in polynomial regression
      - Validation set
        - » To select the model
          - » e.g., the orders in polynomial regression
    - Test set
      - To evaluate the performance

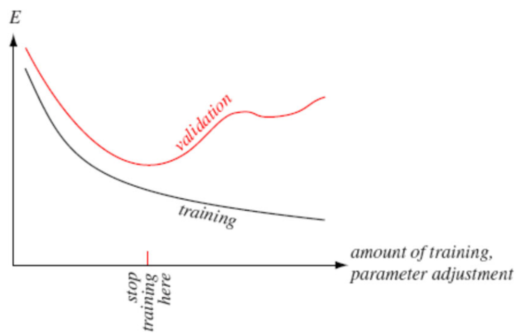


Fig. 9.9 [Duda 01]

Summary

- Independent and identically distributed (i.i.d)
  - The data $D = \{(\mathbf{x}_i, y_i), i = 1, \dots, N\}$ are usually assumed to be iid
    - The ordering is not important
    - All instances are from the same distribution $p(\mathbf{x}, y)$
- Model
  - $g(\mathbf{x}|\theta) \in H$
- Loss function
  - $E(g|D) = E(\theta|D) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, g(\mathbf{x}_i|\theta))$
- Optimization procedure
  - To find θ^* that minimize the total error
    - $\theta^* = \underset{\theta}{\operatorname{argmin}}\, E(\theta|D)$